

ソフトウェアデバッグエキスパートシステム

4G-5

赤尾由香利セシリア 今井恵一 土田賢省

日本電気(株)

1. はじめに

デバッグエキスパートシステム(DBES)は交換系ソフトウェアの試験工程における障害やエラーに対する診断、デバッグを目的とする。本システムは、試験者の自然な思考過程を反映し、さらに知識提供者とシステム利用者の双方の要求を満たすものとするため、専門家と利用者との協力のもとで開発が進められた。デバッグ特有の作業過程を反映するために、DBESでは、初期情報、付加情報、不確定情報等の概念を導入した。また、知識を木で表示したり、推論に使われる事実内容の場合に応じてユーザが変更できる等の機能を備えている。

本稿では、DBESの特徴について、新たに導入した概念やDBESのもつ機能を中心に説明する。

2. 交換系ソフトウェアにおけるデバッグの知識

交換系ソフトウェアにおいては、そのほとんどのソフトが大規模かつ複雑であり、内容は非常に広範な領域の知識を含んでいる。そのため、交換系ソフトの開発においては、特にその試験工程は複雑なものとなり、多大の工数を必要としてきた。また、試験工程では、リアルタイムシステム、OS、ハードウェア等の知識も前提として要求され、経験の浅い者には殆ど対処できない状況である。このようなことから、非熟練者も工数に組み入れ生産性の向上をはかるべく、試験工程を支援するシステムが強く望まれていた。DBESは、これらの要望に答えて開発されたものである。

DBESはプログラムの構文や構造を解析するアプローチを取っていない。その主な理由は、現状の試験工程においてソフトウェアのコーディングレベルの分析は専門家にとっても2次的なものであることと、大規模かつ複雑なソフトでは解析の情報が膨大なものとなり実際のでないことの2つである。DBESでは、対象システムのサービスや機能構成そして各機能間の結合といったレベルの障害やエラーを診断する。そのため、DBESは交換システムのサービスや機能構成そして各機能間の相互作用のどこがおかしいかを特定する調査の知識と特定した不良箇所の修正方法についての知識をもっている。さらにDBESの知識には、対象システムの各機能やサービスについての記述も含まれている。

3. DBESの概要

知識は、ルールの集合という形で蓄えている。これは、デバッグ環境に関する知識を含んでおり、推論はこのルールを用いて進められる。推論に関与しない情報は、これらのルールとは別に蓄えられている。

推論は、ルール駆動型であり、推論に伴い、システムはユーザに質問を繰り返す。各質問には、一連の選択肢が表

示される。各選択肢は、その質問の一つの可能な回答になっている。ユーザがある選択肢を選ぶと、それが作業記憶域に格納され、再び次の質問をしていく。システムが診断を下すのに十分な情報を得たとき、つまり最終結論に達したとき、バグの原因やそのバグの修正法をメッセージとしてユーザに提示する。

DBESの特徴の一つとして通常の推論に加えて、真偽の判然としない情報も扱えるようなメカニズムを備えている。ここではこの情報を不確定情報(unknown information)と呼ぶ。また、この情報を取り扱う機構を、不確定情報を入力したところへフィードバック(feedback)する機構も含めて、unknown treatment mechanismと呼ぶ。

診断中にユーザの助けとなるツールがいろいろ備わっている。ユーザはいつでもこれら呼び出し、使うことができる。以下、簡単に説明する。

- see dynamic memory -全ての得られている動的なデータを表示する。それらは次の1)~5)である。

1) working memory -既出の全事実(known+unknown)。

2) unknown memory -不確定情報(unknown information)。

3) ex_unknown memory -以前不確定であったもの。フィードバックの過程においてunknown memoryから削除された不確定情報が入る。

4) deleted_memory -ユーザにより意図的にworking memoryから取り除かれた情報。

5) initial_memory -初期情報。working memoryの一部。

- take off work memory -working memoryから情報を消去する。これはユーザが自分が誤った選択をしたと判断したときや、推論のパスを故意に変更したいときに使う。

- show conclusions -あるバグが与えられたとき、そのバグの原因として考えられる全ての可能性のある結論を表示する。

- initial information -推論を始める前に、ユーザに既知の情報があればそれらを事実として入力することができる。初期情報(initial information)の内容は途中で変更がない限り、診断中は常に成立する事実として扱われる。

- mark information -推論中に現われたある情報に対してそれが以後の診断において事実として成り立つと判断したときにマークをする。(この情報はinitial informationに追加される。)

- more information -推論には関係しない付加情報を表示する。DBESの知識が専門用語で記述されていたり、交換システムのアーキテクチャの仕様を含んでいるために、ユーザがDBESからのメッセージを理解できない場合など、補足説明などの役割を果たす。

- tree structure -ルールデータベース(static memory)の内容を木構造の形で表示する。

Software Debug Expert System

Cecilia Yukari Akao, Keiichi Imai, Kensei Tsuchida
NEC Corporation

1 呼び出し中のままでない
 ランプの点検
 1.1 ランプが全然ついていない
 NCUの状態番号をダンプリし、着呼回線調べる
 1.1.1 着呼の状態が15
 SCU内のNLCの状態とトレース情報をダンプリ
 1.1.1.1 NLC状態0かつトレース有り
 トレース情報のSIGは着呼通知か?
 1.1.1.1.1 SIGは着呼通知...
 1.1.1.1.2 SIGは着呼通知でない...
 1.1.1.2 NLC状態0かつトレースなし
 NLCの受信処理のSIGNDパラメータをチェック
 1.1.1.2.1 NG...
 1.1.1.2.2 OK...
 1.1.1.3 NLC状態5...
 1.1.1.4 NLC状態6...
 1.1.1.5 NLC状態10, 11...
 1.1.1.6 NLC状態0, 5, 6, 10, 11以外...
 1.1.2 着呼の状態が16...
 1.1.3 着呼の状態が17...
 1.1.4 全回線がALL 0...
 1.1.5 着呼の状態が0, 15, 16, 17以外...
 1.2 HLT ON
 どこでおろしたかをダンプリ情報で調べる
 1.2.1 APL
 ダウンした場所は?
 1.2.1.1 SVCでダウン
 1.2.1.1.1 パラメタリストを...
 1.2.1.2 共通でダウン...
 1.2.1.3 論理でダウン...
 1.2.2 ドライブ...
 1.2.3 OS...

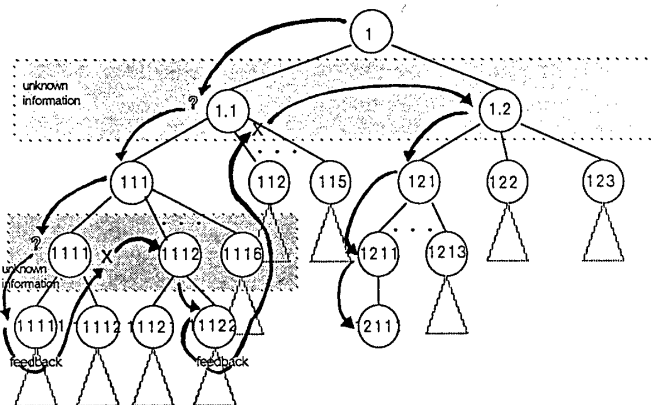


図3. unknown mechanism

図1. 知識の木表現: 各ノードの番号は構造を分かりやすくするために付けたものであり、必ずしも必要でない。

- editor of rules -知識データの追加・修正を行う。これは、診断中に呼び出して使うことも、単独に知識エディタとして使うことも可能である。

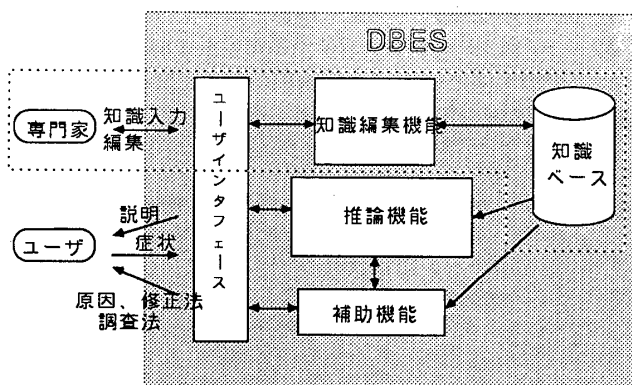


図2: DBESシステム構成

4. 不確定情報の扱い

ユーザがDBESから発せられる質問の返答に窮したとき、DBESではどのように対処するかを説明する。ユーザが質問や選択肢の意味を理解できないとか、more informationなどの補足説明をされてもどれが正しい選択肢なのか判断できない場合などである。このようなとき、ユーザは無理に選択肢の中から質問の答えを選ぶ必要はなく、'unknown'を選べばよい。ここでは、このようにして呼び出された質問を不確定質問(unknown question)と呼ぶ。即ち、'unknown'の指定直後、システムから不確定質問が発せられる。提示する選択肢の中から最も確からしいと思うものを選ぶようにユーザに促してくる。そのときユーザが選んだものが不

確定情報(unknown information)となる。この不確定情報はunknown memoryとworking memoryに蓄えられる。この情報は以後、事実として扱われ推論が続けられる。また、ユーザがどの選択肢が最も確からしいか分からないときは、提示された選択肢は上から下へと確からしい順(専門家の経験による)に並べてあることを参考にできる。

推論が予想外の方向に向かっているとか、そのまま推論を続けても満足できる結論が得られそうもないと判断される場合、それは以前に選んだ不確定情報が不的確なものであったためと考えられる。このような状況に対応するため、不確定質問のやり直しを可能とするfeedback unknown機能を実現した。この機能をユーザが選択すると、入力済の不確定情報に対応する質問が再び発せられる。このとき、ユーザが入力する情報は、また同じ不確定情報である必要はない。情報の再入力後は、その地点から推論が継続される。また、ユーザは何個でも不確定情報を入力できる。少し、詳細に言えば、システムは、不確定情報をスタック形式で扱っており、新たに入力されたものは、unknown memoryの最上部に蓄えられる。feedback unknownを起動したとき、unknown memoryから最後に入れた不確定情報を取り除き、ex_unknown memoryに移す。同時に、その不確定情報と関連している質問を再度行い、推論を続行する。

不確定質問以外の質問に対する回答は確定事実として扱われ、feedback unknownを起動してもworking memoryから取り除かれることはない。

5. 結論

プロトタイプシステムは既に開発を完了し、現在社内でも試験的に使用されている。試行結果から、システムは、交換ソフトのデバッグの分野において、十分使えるものであるとの裏付けを得た。ただし、現プロトタイプシステムが持つ知識はマルチメディアシステムにおけるデバッグの一部に関するものであり、知識の拡張が必要である。また、show conclusionsやルールの木表示などとともに、不確定情報と初期情報の扱いが特に有用であることが立証された。

今後は、実用に向けて、リアルタイムで知識の追加・修正ができ、大量の知識を効率的に扱えるようにDBESの拡充を図るつもりである。そして、特に専門家の知識入力を容易にするために、知識エディタの強化(例、知識のグラフィック表現)に力を入れていきたい。